# Backpropagation: Equations

Ki Hyun Kim

nlp.with.deep.learning@gmail.com

# Recall Objective:

- 주어진 데이터에 대해서 출력 값을 똑같이 내는 함수를 찾고 싶다.
  - 비선형 함수를 통해 더 잘 모사해보자.

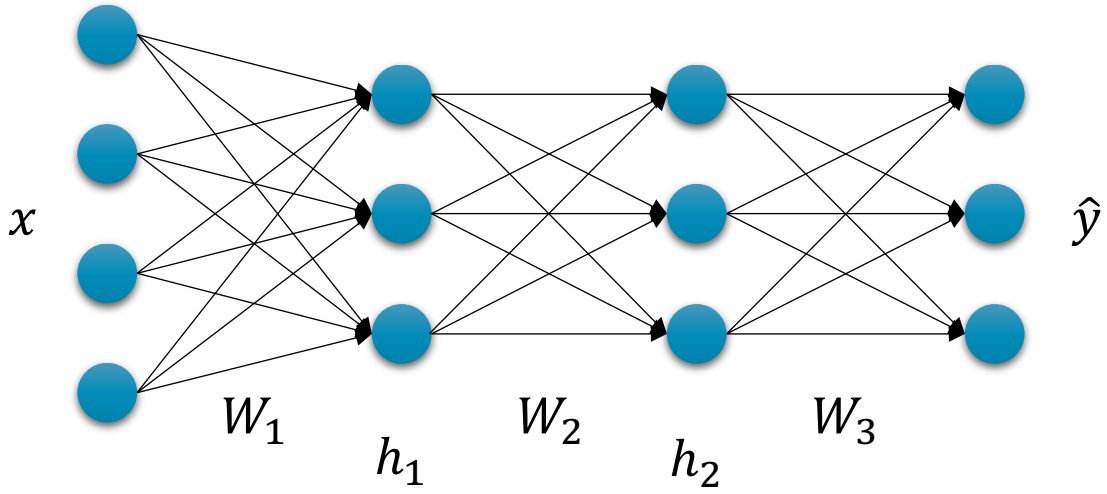
- Loss 값을 최소로 하는 Loss Function의 입력 값($\theta$)을 찾자.

- Gradient Descent를 통해 현재 $\theta_t$에서 더 나은 $\theta_{t+1}$로 나아가자.
  (t 는 iteration의 횟수)

- 그런데 layer가 많아서 미분이 더럽다...

# Without Backpropagation…

- Loss 값을 학습 파라미터마다 미분 해줘야 한다.



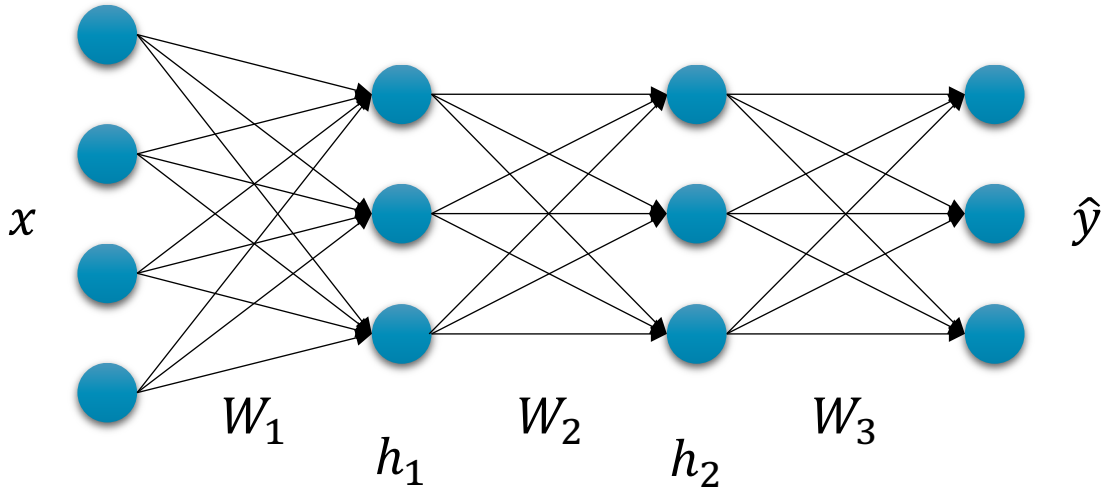

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \|y_i - \hat{y}_i\|_2^2$$

$$\hat{y}_i = h_{2,i} \cdot W_3 + b_3$$
$$h_{2,i} = \sigma(h_{1,i} \cdot W_2 + b_2)$$
$$h_{1,i} = \sigma(x_i \cdot W_1 + b_1)$$

* bias는 편의상 생략하였습니다.

# Without Backpropagation…

- Loss 값을 학습 파라미터마다 미분 해줘야 한다.

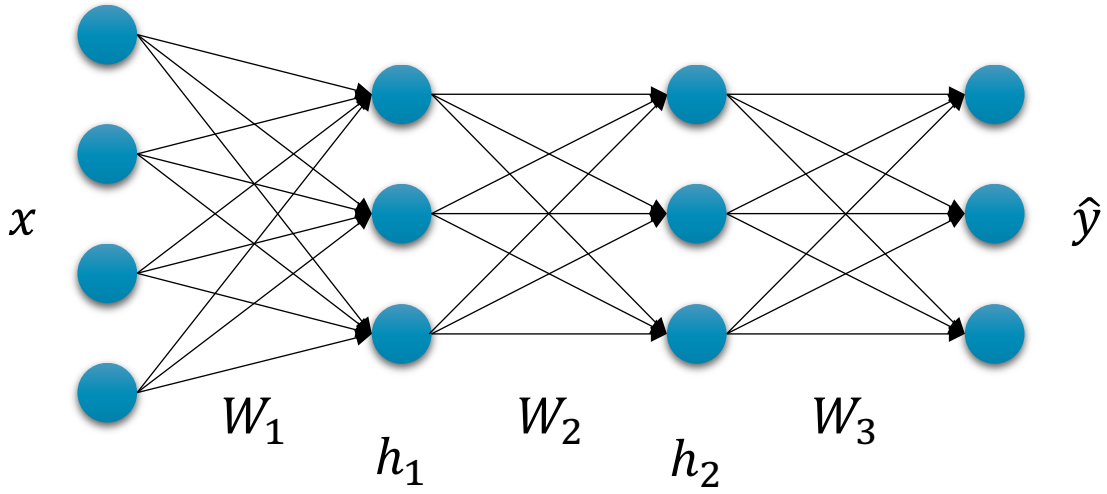



$$W_1 \leftarrow W_1 - \eta \nabla_{W_1} \mathcal{L}(\theta)$$
$$W_2 \leftarrow W_2 - \eta \nabla_{W_2} \mathcal{L}(\theta)$$
$$W_3 \leftarrow W_3 - \eta \nabla_{W_3} \mathcal{L}(\theta)$$

* bias는 편의상 생략하였습니다.

# Without Backpropagation…

- Loss 값을 학습 파라미터마다 미분 해줘야 한다.




$$\nabla_{W_3}\mathcal{L}(\theta) = \nabla_{W_3}\sum_{i=1}^{N}\Big(y_i - (h_{2,i}\cdot W_3 + b_3)\Big)^2$$

$$\nabla_{W_2}\mathcal{L}(\theta) = \nabla_{W_2}\sum_{i=1}^{N}\Big(y - \Big(\sigma(h_{1,i}\cdot W_2 + b_2)\cdot W_3 + b_3\Big)\Big)^2$$

$$\vdots$$

* bias는 편의상 생략하였습니다.

# Backpropagation with Chain Rule

- 매번 처음부터 새로 계산할 필요 없이, 필요한 부분을 재활용

$$\frac{\partial \mathcal{L}}{\partial W_3} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W_3}$$

$$\frac{\partial \mathcal{L}}{\partial W_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial h_2} \cdot \frac{\partial h_2}{\partial W_2}$$

$$\frac{\partial \mathcal{L}}{\partial W_1} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial W_1}$$

# Summary

- Chain rule을 통해, (합성 함수를 전개한) 복잡한 수식에 대한 미분을 각 sub-함수 별 미분의 곱으로 표현 가능함
  - 따라서 각 함수 별 미분을 한 후에 결과 값을 곱해주면 끝

- 물론 실제 PyTorch에선 이러한 미분 과정을 사용자가 직접 수행하지 않음
  - Backward() 함수 호출을 통해 자동으로 미분 결과를 얻을 수 있음